# Dokumentationskopf

**Team:**

5, Marc Kaepke & Constantin Wahl

**Aufgabenteilung und Bearbeitungszeitraum:**

Gemeinsam:

13 Stunden (Implementierung der Algorithmen & Erarbeitung der Skizze)

C. Wahl:

1 Stunde (Messungen und Interpretation)

M. Kaepke

0.5 Stunden (JUnit-Tests)

**Quellenangabe:**

Vorlesungsfolien

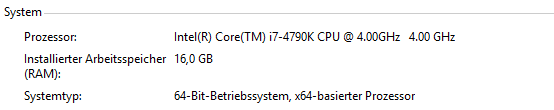
**Aktueller Stand:**

Der InsertionSort und QuickSort Algorithmus sind implementiert und mit JUnit getestet.

Die Messungen und der Vergleich der beiden Algorithmen wurden erstellt (in diesem Dokument ab Seite 2).

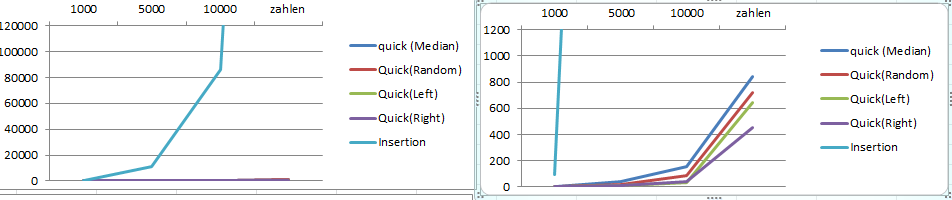
### Versuchsreihe zum Vergleich von Quicksort und Insertionsort

Im Folgenden werden Versuchsdaten zu den beiden Sortieralgorithmen Insertionsort und Quicksort gegenübergestellt. Die Daten wurden an einem Computer mit den unten aufgeführten technischen Daten erhoben.

Technische Daten: 

Die Algorithmen wurden an 4 zufällig erzeugten Zahlenmengen getestet. Der Vergleich listet die Laufzeit, Lese- und Schreibzugriffe auf das Array auf.

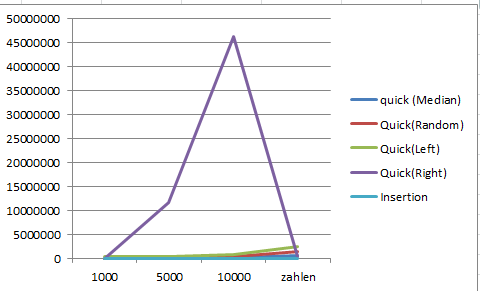
Das Resultat zeigt teilweise relativ deutlich den Charakter der Algorithmen. Allerdings sind manche Verläufe nicht erklärlich. So zum Beispiel der Abfall der Lesezugriffe von 10000 auf 20000 Zahlen bei Quicksort mit dem Pivot „Right“ (in der Grafik „Quick(Right)“).

Runtime: 

*Y-Achse: Laufzeit in ms Dies ist ein Ausschnitt aus der linken Grafik*

*X-Achse: Anzahl der Elemente (zahlen = 20000)*

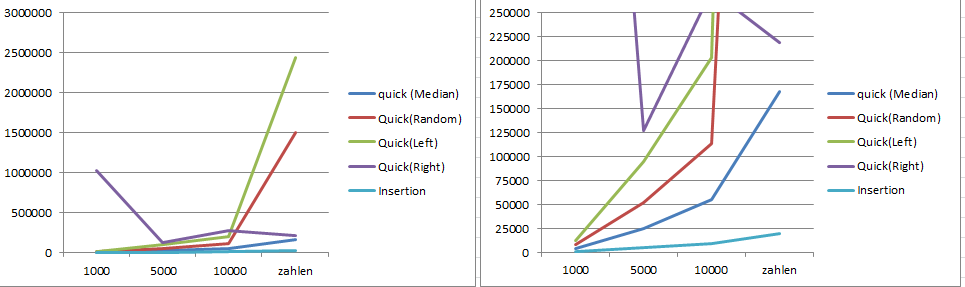
Read:



*Y-Achse: Lesezugriffe*

*X-Achse: Anzahl der Elemente (zahlen = 20000)*

Write:



*Y-Achse: Schreibzugriffe Dies ist ein Ausschnitt aus der linken Grafik*

*X-Achse: Anzahl der Elemente (zahlen = 20000)*

Es ist eine deutliche Tendenz zu sehen, dass die Vorsortierung (im QuickSort) die Laufzeit drastisch verkürzt. Allerdings steigen die Zugriffe auf das Array insbesondere bei den lesenden Bereich stark im Vergleich zu Insertionsort.

Ergebnisse bei zahlen.dat:

|  |  |
| --- | --- |
| InsertionSort | 1.530.204 ms |
| QuickSort mit LEFT | **6.115 ms** |
| QuickSort mit RIGHT | **2.340 ms** |
| QuickSort mit MEDIANOF3 | **2.768 ms** |
| QuickSort mit RANDOM | **3.404 ms** |